L. Fribourg, S. Messika and C. Picaronny

Traces of randomized distributed algorithms as Gibbs fields

Research Report LSV–02–12, Sep. 2002

Laboratoire Spécification et Vérification

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

Ecole Normale Supérieure de Cachan
61, avenue du Président Wilson
94235 Cachan Cedex France

http://www.lsv.ens-cachan.fr/Publis/
Research Report LSV-02-12, Lab. Spécification et Vérification, CNRS & ENS de Cachan, France, Sep. 2002

Traces of Randomized Distributed Algorithms As Gibbs Fields

Laurent Fribourg, Stéphane Messika and Claudine Picaronny

LSV, CNRS & ENS de Cachan,
61 av. du Prés. Wilson,
94235 Cachan cedex, France
{fribourg,messika,picaro}@lsv.ens-cachan.fr

Abstract. Randomized self-stabilizing algorithms are algorithms that, whatever the configuration they start from, reach a set $\mathcal{L}$ of "legitimate configuration" with probability 1. The proof of convergence towards $\mathcal{L}$ is generally done by exhibiting a measure over the set of configurations, that decreases at each step of execution until $\mathcal{L}$ is reached. The discovery of such a measure is generally hard and specific to each algorithm. We propose here a new method of proof, inspired by the notion of Gibbs field used in statistical physics. For each trace of execution, we define a notion of energy and show that, when the energy of the traces reaching $\mathcal{L}$ is "much lower" than the energy of the other ones, the algorithm self-stabilizes. The method is illustrated by an original proof of self-stabilization of Herman's algorithm.

Key-words: Verification of Programs, Self-Stabilization, Ring Networks, Statistical Physics.

1 Introduction

The notion of self-stabilization was introduced in computer science by Dijkstra [4]. An algorithm is self-stabilizing if, whatever the initial configuration it starts from, it reaches within a finite time a set $\mathcal{L}$ of "legitimate" configuration, i.e a configuration satisfying a desired property. Self-stabilizing systems have notably received much attention because they propose an elegant way of solving the problem of fault-tolerance [11]. Randomized algorithms are used in self-stabilization in order to improve the efficiency of deterministic algorithms or overcome some of their limitations in presence of symmetrical topologies (see [5]). With randomized self-stabilizing algorithms, the convergence towards $\mathcal{L}$ is guaranteed only with probability 1.

Proving convergence of randomized self-stabilizing algorithms has always been done so far using the same basic principle: one exhibits a measure over the set of configurations that decreases with non-null probability at each step of execution until $\mathcal{L}$ is reached (see, e.g., [1,5,7]). Finding such a measure often requires however a deep knowledge of the algorithm under study. We propose

here instead a new and general method that is inspired by concepts used in statistical physics.

In statistical physics, many systems can be regarded indeed as self-stabilizing randomized algorithms. For example, Ising system is composed of a grid of ferromagnets that take either 1 or -1 value, according to their orientation. Whatever the original configuration is, the system evolves towards an "equilibrium", which can be seen as the counterpart of the notion of legitimate space. The configurations of these systems generally obey a Gibbs distribution: the number of configurations, ordered by energy, decreases exponentially from the lowest energy to the highest one. Equilibrium configurations correspond to the lowest energy.

Sketch of the paper. Inspired by these principles of statistical physics, we will show here roughly speaking that:

1. The set of traces of computations via a randomized distributed algorithm forms naturally a Markov field, i.e., a random field governed by local interactions.
2. By applying a classical theorem of statistical physics (Hammersley-Clifford), it follows that traces obey a Gibbs distribution.
3. If the energy of the traces reaching $\mathcal{L}$ is "much lower" than the energy of the other ones, then the algorithm is self-stabilizing.

Actually, Hammersley-Clifford cannot be applied directly on the original algorithm $\mathcal{A}$. In order to overcome this difficulty, we construct a variant, called ε-version, to which Hammersley-Clifford applies, and whose convergence entails that of $\mathcal{A}$.

Related work. Principles of statistical physics have previously been used in computer science as an attractive alternative to classical methods. In particular:

- Gibbs (or Boltzmann) distributions have been recently proposed as a framework for the random generation of combinatorial configurations in place of classical recursive methods [6].

- In the domain of image processing, Hammersley-Clifford theorem has been invoked to seek Gibbs distributions that restore given pictures. See [2], [3] p. 268, [12] p.192.

However, such principles will be applied here to the verification of randomized self-stabilizing algorithms, for the first time to our knowledge.

The plan of the paper is as follows. After some preliminaries about randomized distributed algorithms (Section 2), we define the notion of ε-algorithm in order to apply Hammersley-Clifford theorem (section 3). We show that the computation traces of an ε-algorithm form a Markov field (Section 4). In Section 5, we show that the set of traces ordered by levels of energy obeys a Gibbs distribution. We then give a sufficient condition on energies that ensures the convergence of the ε-algorithm (Section 6); the convergence of the original algorithm then follows (Section 7). We conclude in Section 8.

2 Randomized Distributed Algorithms

We focus in this paper on the simple topology of linear networks, i.e. rings of finite-state machines. We assume that the communication between machines is done through the reading of neighbors' state. The following material is adapted from [10]. A *randomized distributed algorithm* is a triple $\mathcal{A} = (N, \xrightarrow{\mathcal{A}}, Q)$ where N is the number of machines of the system, $\xrightarrow{\mathcal{A}}$ is a state transition algorithm, and Q is a finite set of states taken by the machines. For the sake of simplicity, we will suppose henceforth that Q contains only two elements 0 and 1. The expression $\neg q$ stands for 1 (resp. 0) when q is 0 (resp. 1). Calculations on indices i of machines are done modulo N. Each machine M_i can observe the state q_{i-1} of its left neighbor M_{i-1}. The algorithm is *uniform* in the sense that $\xrightarrow{\mathcal{A}}$ and Q are common to every machine. A *configuration* of $\mathcal{A}$ is an N-tuple of machine states; if the current state of machine M_i is $q_i \in Q$, then the configuration of the system is an element $\overline{q} = (q_1, \cdots, q_N)$ of Q^N. The state transition algorithm $\xrightarrow{\mathcal{A}}$ is given as a set of *guarded commands* of the form:

`IF` <guard$_1$> `THEN` <command$_1$>
`IF` <guard$_2$> `THEN` <command$_2$>
...
`IF` <guard$_r$> `THEN` <command$_r$>

For all $1 \leq k \leq r$, $\texttt{guard}_k$ is a predicate of the form $x_{i-1} = \alpha_k \wedge x_i = \beta_k$ with $\alpha_k, \beta_k \in Q$. Guards are mutually exclusive and cover all the possible values of α_k and β_k in Q. So, for all configuration $(q_1 \cdots q_N)$ and all position j $(1 \leq j \leq N)$, there is a unique *enabled command*, i.e. a command of index, say k, such that $x_{j-1} = \alpha_k$ and $x_j = \beta_k$ hold. The execution of a command modifies the state q_i of machine M_i. A command is either a deterministic command of the form $x_i' = \gamma$, or a probabilistic one of the form $x_i' = \gamma$ (resp. $\neg\gamma$) with probability $p > 0$ (resp. $1 - p$). We consider here *synchronous schedulers*: under such a scheduler, a transition leads from $\overline{q} = (q_1, \cdots, q_N)$ to $\overline{q}' = (q_1', \cdots, q_N')$, where q_j' is the result of executing the command enabled at position j. Such a transition is written $\overline{q} \xrightarrow{\mathcal{A}} \overline{q}'$. The probability associated to this transition is written $p(\overline{q} \xrightarrow{\mathcal{A}} \overline{q}')$. We have: $p(\overline{q} \xrightarrow{\mathcal{A}} \overline{q}') = \prod_{j=1}^{N} p_j$,
where p_j is the probability of the command transforming q_j into q_j' ($p_j = 1$ when the command enabled at position j is deterministic).

Example 1. In Herman's mutual exclusion algorithm [9], the scheduler is synchronous, the set of states is $Q = \{0, 1\}$, and the number of machines is odd. The transition system $\xrightarrow{\mathcal{A}}$ is defined in a compact way by, for all $1 \leq i \leq N$:

`IF` $x_i \neq x_{i-1}$ `THEN` $x_i' = \neg x_i$
`IF` $x_i = x_{i-1}$ `THEN` $x_i' = x_i$ (resp. $\neg x_i$) with probability $1/2$,

where the first command stands for:

`IF` $x_i = 0 \wedge x_{i-1} = 1$ `THEN` $x_i' = 1$ $(H1)$
`IF` $x_i = 1 \wedge x_{i-1} = 0$ `THEN` $x_i' = 0$ $(H2)$

and the second command for:

IF $x_i = 0 \wedge x_{i-1} = 0$ THEN $x'_i = 0$ (resp. 1) with probability 1/2 $\quad (H3)$

IF $x_i = 1 \wedge x_{i-1} = 1$ THEN $x'_i = 0$ (resp. 1) with probability 1/2. $\quad (H4)$

It is easy to see that, for any configuration $\overline{q}$ and any position i ($1 \leq i \leq N$), there is a unique enabled command at position i. The command is probabilistic when $q_i = q_{i-1}$. For example, consider the configuration $\overline{q} = 10\mathbf{000}$. Letters in bold indicate that they are subjet to randomized transitions. We have $\overline{q} \xrightarrow{\mathcal{A}} \overline{q}'$ with probability $(1/2)^3$, where $\overline{q}'$ is 01110.

Without loss of understanding, we will abbreviate henceforth $\mathcal{A} : (N, \xrightarrow{\mathcal{A}}, Q)$ as $\xrightarrow{\mathcal{A}}$. An *$\mathcal{A}$-trace* $\mathcal{T}$ of length m is a sequence of configurations $T_0, T_1, \cdots, T_m$ in Q^N such that T_0 is the initial configuration and $T_{l-1} \xrightarrow{\mathcal{A}} T_l$ for all $1 \leq l \leq m$. Such a trace can be represented as a matrix of size $(m+1) \times N$, where line 0 corresponds to the initial configuration T_0, and (l, i)-th coordinate ($1 \leq l \leq m$, $1 \leq i \leq N$) corresponds to the state of the i-th machine after l-th step.

Let $S = \{0, 1, \cdots, m\} \times \{1, \cdots, N\}$ be the set of coordinates (or *sites*) of the form (l, i) of $\mathcal{T}$. Let S^0 be the subset of sites of the form $(0, i)$ and S^* the complementary set of sites (l, i) with $l \neq 0$ (i.e. $S = S^0 \uplus S^*$). Expression $\mathcal{T}_{l,i}$ represents the value of the (l, i)-th coordinate of $\mathcal{T}$.

Given an initial configuration T_0, we now define the probability distribution of the random field of traces starting with T_0.

Definition 2. Given T_0, the *probability of an $\mathcal{A}$-trace* $\mathcal{T} = T_0 T_1 \cdots T_m$, written $P(\mathcal{T})$, is $\prod_{l=1}^{m} p(T_{l-1} \xrightarrow{\mathcal{A}} T_l)$.

We are interested in proving the following *convergence* property towards a set $\mathcal{L}$ of legitimate configurations: No matter which initial configuration one starts from, the probability that $\xrightarrow{\mathcal{A}}$ reaches a configuration of $\mathcal{L}$ in a finite number of transitions is 1. Formally:

Definition 3. An $\mathcal{A}$-trace $\mathcal{T} = T_0 T_1 \cdots T_m$ is $\mathcal{L}^{ok}$ if its two last elements T_{m-1} and T_m belong to $\mathcal{L}$. It is $\mathcal{L}^{\neg ok}$ otherwise.

Let $\mathcal{U}_m^{ok}(\mathcal{A}, T_0)$ (resp. $\mathcal{U}_m^{\neg ok}(\mathcal{A}, T_0)$) be the set of $\mathcal{L}^{ok}$ (resp. $\mathcal{L}^{\neg ok}$) $\mathcal{A}$-traces of length m, and 1st element T_0. Clearly, we have:
$\sum_{\mathcal{T} \in \mathcal{U}_m^{ok}(\mathcal{A}, T_0)} P(\mathcal{T}) + \sum_{\mathcal{T} \in \mathcal{U}_m^{\neg ok}(\mathcal{A}, T_0)} P(\mathcal{T}) = 1$.

Definition 4. A randomized algorithm $\mathcal{A}$ is *convergent towards* $\mathcal{L}$ iff, for all $T_0 \in Q^N$:
$\lim_{m \to \infty} \sum_{\mathcal{T} \in \mathcal{U}_m^{ok}(\mathcal{A}, T_0)} P(\mathcal{T}) = 1$, or $\lim_{m \to \infty} \sum_{\mathcal{T} \in \mathcal{U}_m^{\neg ok}(\mathcal{A}, T_0)} P(\mathcal{T}) = 0$.

Definition 5. A randomized algorithm $\mathcal{A}$ is *self-stabilizing w.r.t.* $\mathcal{L}$ iff it is convergent towards $\mathcal{L}$ and $\mathcal{L}$ is closed via $\mathcal{A}$ (i.e.: $(x \in \mathcal{L} \wedge x \xrightarrow{\mathcal{A}} x') \Rightarrow x' \in \mathcal{L}$).

Example 6. In Herman's example, the set $\mathcal{L}$ is the set of configurations with only one token, where a token is defined by the presence of two contiguous states of the same value (0 or 1). It is easy to show that $\mathcal{L}$ is closed. Suppose that $N = 5$ and $m = 3$. The following matrix

$$\begin{pmatrix} 1\,0\,0\,0\,0 \\ 0\,1\,1\,1\,0 \\ 1\,0\,1\,0\,1 \\ 0\,1\,0\,1\,0 \end{pmatrix}$$

represents trace $\mathcal{T} = T_0T_1T_2T_3$ in which T_l corresponds to line l of the matrix ($0 \leq l \leq 3$). Lines 0 and 1 contain three tokens; lines 2 and 3 contain only one token, and belong to $\mathcal{L}$. So $\mathcal{T}$ is an $\mathcal{L}^{ok}$ trace generated by Herman. The probability of going from line 0 to 1 (resp. from 1 to 2) is $(1/2)^3$. The probability of going from line 2 to 3 is $1/2$. Therefore $P(\mathcal{T}) = (1/2)^3(1/2)^3 1/2 = 2^{-7}$.

3 ε-versions of randomized algorithms

Hammersley-Clifford theorem states that, under a certain condition of "positivity", every Markov random field is a Gibbs field (see [3], p.262). Positivity condition ([3], p. 255) means in our context that any configuration is reachable from any other one. This condition is not always met by a given algorithm $\mathcal{A}$ because of the deterministic commands it may contain. However, it is easy to derive from $\mathcal{A}$ a modified algorithm that meets the positivity condition: It suffices to replace every deterministic command of $\mathcal{A}$ of the form

$\texttt{IF } x_{i-1} = \alpha \wedge x_i = \beta \texttt{ THEN } x'_i = \gamma$

with the two following commands:

$\texttt{IF } x_{i-1} = \alpha \wedge x_i = \beta \texttt{ THEN } x'_i = \gamma$ with probability $1 - \varepsilon_l$

$\texttt{IF } x_{i-1} = \alpha \wedge x_i = \beta \texttt{ THEN } x'_i = \neg\gamma$ with probability ε_l,

where ε_l is a real in $]0, 1[$ that depends on the l-th step at which the transition is applied. Given the original algorithm $\mathcal{A}$ and a sequence $\varepsilon = \{\varepsilon_1, \varepsilon_2, \cdots\}$ of reals in $]0, 1[$, the above modified algorithm is called the *ε-version* of $\mathcal{A}$ and written $\mathcal{A}^\varepsilon$. An *ε-algorithm* is the ε-version of some randomized distributed algorithm.

Remark 7. Given an $\mathcal{A}^\varepsilon$-trace $T_0 \cdots T_m$, the probability $p(T_{l-1} \xrightarrow{\mathcal{A}^\varepsilon} T_l)$ of going from T_{l-1} to T_l via the ε-algorithm (for some $1 \leq l \leq m$), depends not only on the form of T_{l-1} and T_l, as in the original algorithm, but also on the value l of the execution step.

Example 8. In Herman's example, randomized commands $(H3)$ and $(H4)$ of the original algorithm are kept unchanged while deterministic commands $(H1)$ and $(H2)$ become, at l-th step, of the form:

$\texttt{IF } x_i = 0 \wedge x_{i-1} = 1 \texttt{ THEN } x'_i = 1$ with probability $1 - \varepsilon_l$ $\quad (H'1)$

$\texttt{IF } x_i = 0 \wedge x_{i-1} = 1 \texttt{ THEN } x'_i = 0$ with probability ε_l $\quad (H''1)$

$\texttt{IF } x_i = 1 \wedge x_{i-1} = 0 \texttt{ THEN } x'_i = 0$ with probability $1 - \varepsilon_l$. $\quad (H'2)$

$\texttt{IF } x_i = 1 \wedge x_{i-1} = 0 \texttt{ THEN } x'_i = 1$ with probability ε_l. $\quad (H''2)$

Given an ε-algorithm $\mathcal{A}^\varepsilon$, the notions of Section 2 are adapted as follows:

Definition 9.

- Given T_0, the *probability of and an $\mathcal{A}^\varepsilon$-trace* $\mathcal{T} = T_0T_1 \cdots T_{m-1}T_m$, written $P(\mathcal{T})$, is $\prod_{l=1}^{m} p(T_{l-1} \xrightarrow{\mathcal{A}^\varepsilon} T_l)$.

- $\mathcal{T}$ is $\mathcal{L}^{ok}$ if T_{m-1} and T_m belong to $\mathcal{L}$ and there exists besides a transition from T_{m-1} to T_m via the original algorithm $\mathcal{A}$ (i.e.: $T_{m-1} \xrightarrow{\mathcal{A}} T_m$).
  It is $\mathcal{L}^{\neg ok}$ otherwise.
- $\mathcal{U}_m^{ok}(\mathcal{A}^\varepsilon, T_0)$ (resp. $\mathcal{U}_m^{\neg ok}(\mathcal{A}^\varepsilon, T_0)$) denotes the set of $\mathcal{A}^\varepsilon$-traces of length m and 1st element T_0, which are $\mathcal{L}^{ok}$ (resp. $\mathcal{L}^{\neg ok}$).
- An ε-algorithm $\mathcal{A}^\varepsilon$ is *convergent* iff, for all $T_0 \in Q^N$:
  $\lim_{m\to\infty} \sum_{\mathcal{T}\in\mathcal{U}_m^{\neg ok}(\mathcal{A}^\varepsilon,T_0)} P(\mathcal{T}) = 0$ or $\lim_{m\to\infty} \sum_{\mathcal{T}\in\mathcal{U}_m^{ok}(\mathcal{A}^\varepsilon,T_0)} P(\mathcal{T}) = 1$.

From these definitions, it easily follows:

Proposition 10. *Given an algorithm $\mathcal{A}$ of ε-version $\mathcal{A}^\varepsilon$ and an initial configuration $T_0 \in Q^N$, we have:* $\mathcal{U}_m^{\neg ok}(\mathcal{A}, T_0) \subseteq \mathcal{U}_m^{\neg ok}(\mathcal{A}^\varepsilon, T_0)$.

Example 11. Consider an $\mathcal{A}^\varepsilon$-trace $\mathcal{T} = T_0T_1T_2T_3$ of the same form of the $\mathcal{A}$-trace of example 6:

$$\begin{pmatrix} 1\,0\,0\,0\,0 \\ 0\,1\,1\,1\,0 \\ 1\,0\,1\,0\,1 \\ 0\,1\,0\,1\,0 \end{pmatrix}$$

$\mathcal{T}$ is an $\mathcal{L}^{ok}$ trace generated by the ε-version of Herman. The probability of going from line 0 to 1 is now $1/8(1-\varepsilon_1)^2$. Likewise, the probability from line 1 to 2 is $1/8(1-\varepsilon_2)^3$, and the probability from line 2 to 3 is $1/2(1-\varepsilon_3)^4$. Therefore, in the ε-version of Herman, $P(\mathcal{T}) = 2^{-7}(1-\varepsilon_1)^2(1-\varepsilon_2)^2(1-\varepsilon_3)^4$.
Note that the following $\mathcal{A}^\varepsilon$-trace

$$\mathcal{T}' = \begin{pmatrix} 1\,0\,0\,0\,0 \\ 0\,1\,1\,1\,0 \\ 1\,0\,1\,0\,1 \\ 1\,0\,1\,0\,1 \end{pmatrix}$$

is *not* $\mathcal{L}^{ok}$ although the two last lines T_2 and T_3' belong to $\mathcal{L}$ because there is no transition via $\mathcal{A}$ from T_2 to T_3'.

4 Computation Traces As Markov Fields

In this section as well as in Sections 5 and 6, the algorithm considered implicitly is always an ε-algorithm $\mathcal{A}^\varepsilon$. We assume that N is given as well as an initial configuration $T_0 \in Q^N$. We first observe:

Proposition 12. *The traces generated by $\mathcal{A}^\varepsilon$ from T_0 form a* random field *on S with values in Q, i.e., a collection $X = \{X_{l,i}\}_{(l,i)\in S}$ of random variables $X_{l,i}$ with values in Q.*

This random field can be regarded as a random variable taking its values in the set of traces Q^S. For a given subset $A \subset S$, define the restriction of X to A as:

$X(A) = \{X_{l,i}\}_{(l,i)\in A}$.

For $l = 0$, the value of random variable $X_{l,i}$ is fixed and equal to the i-th component of T_0 with probability 1. For $l > 0$, the value of random variable $X_{l,i}$ is the state of machine M_i after step l. It depends on the values of $X_{l-1,i}$ and $X_{l-1,i-1}$, i.e. the states of M_i and M_{i-1} just before step l. This means that the field of traces generated by $\mathcal{A}^\varepsilon$ from T_0 is a *Markov field*, i.e. a random field where the distribution of the value of site (l,i) is influenced only by the values of the "neighboring" sites $(l-1,i)$ and $(l-1,i-1)$. Formally, we have the following definitions and proposition:

Definition 13 (Neighbors and cliques). A *neighborhood system* on S is a family $\{\Lambda_{l,i}\}_{(l,i)\in S}$ of subsets of S such that for all $(l,i) \in S$:

1. $(l,i) \notin \Lambda_{l,i}$
2. $(l',i') \in \Lambda_{l,i} \Rightarrow (l,i) \in \Lambda_{l',i'}$.

The subset $\Lambda_{l,i}$ is called the *neighborhood* of site (l,i). (An element of $\Lambda_{l,i}$ is a *neighbor* of (l,i).) *Cliques* are defined as follows:

- any singleton (l,i) is a clique,
- a subset $C \subset S$ with more than one element is a clique iff two distinct sites of C are mutual neighbors.

Definition 14 (Markov Field). The random field X is a *Markov field* with respect to a neighborhood system Λ if for all $(l,i) \in S$ the random variables $X_{l,i}$ and $X(S - (\Lambda_{l,i} \cup (l,i)))$ are independent given $X(\Lambda_{l,i})$.

Proposition 15. *The random field X of traces generated by ε-algorithm $\mathcal{A}^\varepsilon$ from a given initial configuration T_0, is a Markov field whose cliques are, for all $s = (l,i) \in S$, of the form C_s^1, C_s^2 or C_s^3 where:*

- *C_s^1 is singleton $\{(l,i)\}$,*
- *C_s^2 is pair $\{(l,i),(l-1,i)\}$ for $s \in S^*$ (i.e., $l \neq 0$),*
- *C_s^3 is pair $\{(l,i),(l-1,i-1)\}$ for $s \in S^*$ (i.e., $l \neq 0$).*

The form of cliques C_s^1, C_s^2 and C_s^3 appearing in this proposition is explained in appendix A. For $(l,i) \in S^*$, let us write $\Pi_{l,i}\begin{pmatrix} \alpha & \beta \\ \cdot & \gamma \end{pmatrix}$ the conditional probability that the value of site (l,i) be γ, when the values of sites $(l-1,i-1)$ and $(l-1,i)$ are α and β respectively. This is called the *local characteristic* at site (l,i) for $(\alpha,\beta,\gamma) \in Q^3$. We have:

Proposition 16. *For all $(l,i) \in S^*$ and $\alpha,\beta,\gamma \in Q$, we have: $\Pi_{l,i}\begin{pmatrix} \alpha & \beta \\ \cdot & \gamma \end{pmatrix} = p$, where p is the probability of executing at step l the command of $\mathcal{A}^\varepsilon$:*

`IF` $x_{i-1} = \alpha \wedge x_i = \beta$ `THEN` $x_i' = \gamma$.

The local characteristics at site $(l,i) \in S^*$ generally depend on index l (cf. remark 7). On the other hand, they do not depend on i since $\mathcal{A}$ (hence $\mathcal{A}^\varepsilon$) is uniform.

Example 17. For the ε-version of Herman, local characteristics at site $(l,i) \in S^*$, are as follows:

$$\Pi_{l,i}\begin{pmatrix} 0\ 1 \\ .\ 0 \end{pmatrix} = 1-\varepsilon_l \qquad \text{(corresponding to command } (H'1))$$

$$\Pi_{l,i}\begin{pmatrix} 0\ 1 \\ .\ 1 \end{pmatrix} = \varepsilon_l \qquad \text{(corresponding to command } (H''1))$$

$$\Pi_{l,i}\begin{pmatrix} 1\ 0 \\ .\ 0 \end{pmatrix} = \varepsilon_l \qquad \text{(corresponding to command } (H'2))$$

$$\Pi_{l,i}\begin{pmatrix} 1\ 0 \\ .\ 1 \end{pmatrix} = 1-\varepsilon_l \qquad \text{(corresponding to command } (H''2))$$

$$\Pi_{l,i}\begin{pmatrix} 0\ 0 \\ .\ 0 \end{pmatrix} = 1/2 \qquad \text{(corresponding to command } (H3))$$

$$\Pi_{l,i}\begin{pmatrix} 0\ 0 \\ .\ 1 \end{pmatrix} = 1/2 \qquad \text{(corresponding to command } (H3))$$

$$\Pi_{l,i}\begin{pmatrix} 1\ 1 \\ .\ 0 \end{pmatrix} = 1/2 \qquad \text{(corresponding to command } (H4))$$

$$\Pi_{l,i}\begin{pmatrix} 1\ 1 \\ .\ 1 \end{pmatrix} = 1/2 \qquad \text{(corresponding to command } (H4))$$

5 Computation Traces As Gibbs Fields

We can now define the notion of energy deriving from the local characteristics of the Markov field of traces. We then show that the set of traces ordered by levels of energy obey a Gibbs distribution.

5.1 Energy of Trace

We have seen that the traces generated by an ε-algorithm $\mathcal{A}^\varepsilon$ starting from a given configuration T_0, define a Markov field. For each site $s \in S$ and each clique C_s^1 (resp. C_s^2, C_s^3) of the field, one can then associate a function $V_{C_s^1}$ (resp. $V_{C_s^2}$, $V_{C_s^3}$), called "clique potential", that maps traces starting with T_0 into reals. The clique potentials derive from the local characteristics of the Markov field: see appendix B. It is convenient to group potentials $V_{C_s^1}$, $V_{C_s^2}$ and $V_{C_s^3}$ together. We call *local potential at* $s = (l,i) \in S^*$, and write W_s, the expression:

$$W_s = V_{C_s^1} + V_{C_s^2} + V_{C_s^3}.$$

For any trace $\mathcal{T}$ and any site $s = (l,i) \in S^*$, the local potential $W_s(\mathcal{T})$ will be written sometimes $W_s\begin{pmatrix} \alpha\ \beta \\ .\ \gamma \end{pmatrix}$ with $\alpha = \mathcal{T}_{l-1,i-1}$, $\beta = \mathcal{T}_{l-1,i}$ and $\gamma = \mathcal{T}_{l,i}$.

The *transition energy from configuration* T_{l-1} *to* T_l, written $E(T_{l-1} \to T_l)$ is the sum of potentials $W_{l,i}$ when $l > 0$ is fixed:

$$E(T_{l-1} \to T_l) = \sum_{1 \leq i \leq N} \ W_{l,i}(\mathcal{T}).$$

The *energy* $E(\mathcal{T})$ *of trace* $\mathcal{T}$=$T_0 \cdots T_m$ is the sum of the transition energies:

$$E(\mathcal{T}) = \sum_{1 \leq l \leq m} E(T_{l-1} \to T_l) \qquad (= \sum_{(l,i) \in S^*} \ W_{l,i}(\mathcal{T})).$$

Let us finally write $Max(\mathcal{E}_m^{ok})$ the maximal energy for all the $\mathcal{L}^{ok}$ traces of length m and all the possible starting configurations, i.e.:

$$Max(\mathcal{E}_m^{ok}) = sup\{E(\mathcal{T}) \mid \mathcal{T} \in \mathcal{U}_m^{ok}(\mathcal{A}^{\varepsilon}, I) \text{ for some } I \in Q^N\}.$$

Likewise, let us write $Min(\mathcal{E}_m^{\neg ok})$ the minimal energy for all the $\mathcal{L}^{ok}$ traces of length m and all the possible starting configurations, i.e.:

$$Min(\mathcal{E}_m^{\neg ok}) = inf\{E(\mathcal{T}) \mid \mathcal{T} \in \mathcal{U}_m^{\neg ok}(\mathcal{A}^{\varepsilon}, I) \text{ for some } I \in Q^N\}.$$

Such extreme energies will play a basic role for proving the convergence of $\mathcal{A}^{\varepsilon}$ (see Section 6). In the following, we compute the extreme energies in the case of Herman's algorithm.

5.2 Extreme Energies for Herman's algorithm

Let ε be some sequence of reals in $]0,1[$. Let us express $Max(\mathcal{E}_m^{ok})$ and $Min(\mathcal{E}_m^{\neg ok})$ for the ε-version of Herman's algorithm. Let $\lambda_l = \varepsilon_l/(1-\varepsilon_l)$, for all $1 \leq l \leq m$. We have, using the values of the local characteristics found in example 17, the following local potentials $W_{l,i}$ (see appendix B):

$$W_{l,i}\begin{pmatrix} 0\ 1 \\ .\ 1 \end{pmatrix} = -ln(\lambda_l) - ln(2)$$

$$W_{l,i}\begin{pmatrix} 1\ 0 \\ .\ 1 \end{pmatrix} = ln(\lambda_l) - ln(2)$$

$$W_{l,i}\begin{pmatrix} \alpha\ \beta \\ .\ \gamma \end{pmatrix} = -ln(2) \qquad \text{for all the other triples } (\alpha, \beta, \gamma) \in Q^3.$$

Observe that there are only three values for these potentials: $-ln(\lambda_l) - ln(2), ln(\lambda_l) - ln(2)$ and $-ln(2)$. One can always assume that ε_l (hence λ_l) is "very small", which yields: $-ln(\lambda_l) - ln(2) \gg 0 > -ln(2) \gg ln(\lambda_l) - ln(2)$.

Energy $Max(\mathcal{E}_m^{ok})$.
At lines $m-1$ and m, the form of an $\mathcal{L}^{ok}$ trace is determined, of the form:

$$\begin{pmatrix} 0\ 1\ 0\ 1\ 0 \cdots \\ 1\ 0\ 1\ 0\ 1 \cdots \end{pmatrix}$$

Before line $m-1$, we can assume that the trace has a maximal energy: it contains a maximum of local potentials of value $-ln(\lambda_l)$ and a minimum of potentials of values $ln(\lambda_l)$, i.e., a maximum of patterns $\begin{pmatrix} 0\ 1 \\ .\ 1 \end{pmatrix}$ and a minimum of $\begin{pmatrix} 1\ 0 \\ .\ 1 \end{pmatrix}$.

This happens when lines $0, 1, \cdots, m-2$ are identical of the form:

$$\begin{pmatrix} 0\,1 & 0 & 1\,0\cdots \\ 0\,1 & 0 & 1\,0\cdots \\ & \cdots & \\ 0\,1 & 0 & 1\,0\cdots \end{pmatrix}$$

The $\mathcal{L}^{ok}$ trace of maximal energy is thus of the form:

$$\begin{pmatrix} 0\,1 & 0 & 1\,0\cdots \\ 0\,1 & 0 & 1\,0\cdots \\ & \cdots & \\ 0\,1 & 0 & 1\,0\cdots \\ 1\,0 & 1 & 0\,1\cdots \end{pmatrix}$$

We have:

$E(T_{l-1} \to T_l) = -\frac{N-1}{2} ln(\lambda_l) - N ln(2)$, for all $1 \leq l \leq m-1$,

$E(T_{m-1} \to T_m) = \frac{N-1}{2} ln(\lambda_m) - N ln(2)$.

The maximal energy of an $\mathcal{L}^{ok}$ trace is obtained by summation:

$$Max(\mathcal{E}_m^{ok}) = \frac{N-1}{2} ln(\lambda_m) - \frac{N-1}{2} \sum_{l=1}^{m-1} ln(\lambda_l) - mN ln(2).$$

Energy $Min(\mathcal{E}_m^{\neg ok})$.

The minimal energy of an $\mathcal{L}^{\neg ok}$ trace is:

$Min(\mathcal{E}_m^{\neg ok}) = \frac{N-3}{2} ln(\lambda_m) + \frac{N-1}{2} \sum_{l=1}^{m-1} ln(\lambda_l) - (mN+1) ln(2)$.

This is explained in appendix C.

5.3 Application of Hammersley-Clifford

Let us now apply Hammersley-Clifford theorem for proving that the Markov field of traces is a Gibbs field. We first define the notion of Gibbs field in our context.

Definition 18 (Gibbs Field). The random field of traces generated by an ε-algorithm starting from a given configuration T_0, is a *Gibbs field* if its probability distribution P is such that, for all trace $\mathcal{T}$ starting with T_0:

$$P(\mathcal{T}) = 1/Z \; e^{-E(\mathcal{T})}$$

where E is the energy defined in Section 5.1 and Z a normalizing constant.

For ε-algorithms, we know that the set of traces of computations starting from a given T_0 is a Markov field (see Proposition 15). Furthermore the positivity condition is met (see Section 3). Now Hammersley-Clifford theorem states that, under this condition, a Markov field X is a Gibbs field for the energy E deriving from the local characteristics of X (see [3], p.262). We have:

Theorem 19. *The field of traces generated by application of ε-algorithm $\mathcal{A}^{\varepsilon}$ starting from a given T_0, is a Gibbs field (i.e.: $P(\mathcal{T}) = 1/Z \; e^{-E(\mathcal{T})}$).*

The trace of maximal (resp. minimal) probability thus corresponds to a minimal (resp. maximal) energy.

6 A Criterion of Convergence for the ε-Algorithm

Using the fact that, given T_0, an ε-algorithm $\mathcal{A}^\varepsilon$ defines a Gibbs field (see Theorem 19), we give a sufficient condition on the extreme energies for ensuring the convergence of $\mathcal{A}^\varepsilon$. In this section, $\mathcal{U}_m^{\neg ok}(T_0)$ abbreviates for $\mathcal{U}_m^{\neg ok}(\mathcal{A}^\varepsilon, T_0)$.

Theorem 20. *If:* $\lim_{m\to\infty} Max(\mathcal{E}_m^{ok}) - Min(\mathcal{E}_m^{\neg ok}) + mNln(2) = -\infty$, *then $\mathcal{A}^\varepsilon$ is convergent.*

Proof:
In order to prove the convergence of $\mathcal{A}^\varepsilon$, it suffices to show that, for all $T_0 \in Q^N$: $\lim_{m\to\infty} \sum_{\mathcal{T}\in\mathcal{U}_m^{\neg ok}(T_0)} P(\mathcal{T}) = 0$.

Alternatively, it suffices to prove: $\lim_{m\to\infty} \dfrac{\sum_{\mathcal{T}\in\mathcal{U}_m^{\neg ok}(T_0)} P(\mathcal{T})}{\sum_{\mathcal{T}\in\mathcal{U}_m^{ok}(T_0)} P(\mathcal{T})} = 0.$

Consider the numerator. We have:

$$\sum_{\mathcal{T}\in\mathcal{U}_m^{\neg ok}(T_0)} P(\mathcal{T}) \leq \frac{2^{mN}}{Z} e^{-Min(\mathcal{E}_m^{\neg ok})}$$

because there are at most 2^{mN} $\mathcal{L}^{\neg ok}$ traces of size m and each of them has an energy greater than or equal to $Min(\mathcal{E}_m^{\neg ok})$. As for the denominator, we have:

$$\sum_{\mathcal{T}\in\mathcal{U}_m^{ok}(T_0)} P(\mathcal{T}) \geq \frac{1}{Z} e^{-Max(\mathcal{E}_m^{ok})}$$

because

$$\sum_{\mathcal{T}\in\mathcal{U}_m^{ok}(T_0)} P(\mathcal{T}) \;\geq\; Max\{P(\mathcal{T}) \mid \mathcal{T}\in\mathcal{U}_m^{ok}(T_0)\} \;=\; \frac{1}{Z} e^{-Max\{E(\mathcal{T}) \mid \mathcal{T}\in\mathcal{U}_m^{ok}(T_0)\}} \;\geq\; \frac{1}{Z} e^{-Max(\mathcal{E}_m^{ok})}.$$

Therefore: $\dfrac{\sum_{\mathcal{T}\in\mathcal{U}_m^{\neg ok}(T_0)} P(\mathcal{T})}{\sum_{\mathcal{T}\in\mathcal{U}_m^{ok}(T_0)} P(\mathcal{T})} \;\leq\; 2^{mN}\, \dfrac{e^{-Min(\mathcal{E}_m^{\neg ok})}}{e^{-Max(\mathcal{E}_m^{ok})}}.$

In order to prove the convergence of $\mathcal{A}^\varepsilon$, it suffices to show that the left part tends towards 0. So it suffices to show that the right part tends towards 0, or its logarithm towards $-\infty$, i.e.:

$$\lim_{m\to\infty} Max(\mathcal{E}_m^{ok}) - Min(\mathcal{E}_m^{\neg ok}) + mNln(2) = -\infty.$$

□

It follows:

Theorem 21. *For any sequence ε decreasing fast enough, the ε-version of Herman's algorithm is convergent.*

Proof:
Let us recall $Max(\mathcal{E}_m^{ok})$ and $Min(\mathcal{E}_m^{\neg ok})$ for Herman (see Section 5.2):

$Max(\mathcal{E}_m^{ok}) = \frac{N-1}{2} ln(\lambda_m) - \frac{N-1}{2} \sum_{i=1}^{m-1} ln(\lambda_i) - mNln(2)$.

$Min(\mathcal{E}_m^{\neg ok}) = \frac{N-3}{2} ln(\lambda_m) + \frac{N-1}{2} \sum_{i=1}^{m-1} ln(\lambda_i) - (mN+1)ln(2)$.

Hence:

$Max(\mathcal{E}_m^{ok}) - Min(\mathcal{E}_m^{\neg ok}) + mNln(2) = ln\lambda_m - (N-1) \sum_{l=1}^{m-1} ln\lambda_l + (mN+1)ln(2)$.

It suffices to take ε_m (hence λ_m) as decreasing fast enough so that the right expression tend towards $-\infty$ with m (e.g.: $\varepsilon_m = 1/2^{2^{2^m}}$). It then follows from theorem 20 that the ε-version of Herman is convergent. □

7 Application to the Proof of Self-Stabilization

Let us now show that convergence of $\mathcal{A}^\varepsilon$ entails the convergence of $\mathcal{A}$.

Theorem 22. *If, for some sequence ε, the ε-algorithm $\mathcal{A}^\varepsilon$ is convergent, then $\mathcal{A}$ is convergent.*

Proof:
Let T_0 be a given starting configuration, m a natural number and $\mathcal{T}$ a trace of $\mathcal{U}_m^{\neg ok}(\mathcal{A}, T_0)$. We know that $\mathcal{T}$ also belongs to $\mathcal{U}_m^{\neg ok}(\mathcal{A}^\varepsilon, T_0)$ (since $\mathcal{U}_m^{\neg ok}(\mathcal{A}, T_0) \subseteq \mathcal{U}_m^{\neg ok}(\mathcal{A}^\varepsilon, T_0)$, by Proposition 10). Let $P^0(\mathcal{T})$ (resp. $P^\varepsilon(\mathcal{T})$) be the probability of $\mathcal{T}$ for algorithm $\mathcal{A}$ (resp. $\mathcal{A}^\varepsilon$). We have:

$$P^0(\mathcal{T}) \leq \frac{P^\varepsilon(\mathcal{T})}{(1-\varepsilon_1)^N \cdots (1-\varepsilon_m)^N}$$

This is because, the difference between P^0 and P^ε is the largest when computation via $\mathcal{A}$ uses N deterministic commands at each step $1, \cdots, m$. Now, one can always assume that sequence $\{\varepsilon_l\}$ decreases fast enough so that product $\Delta_m = (1-\varepsilon_1)^N \cdots (1-\varepsilon_m)^N$ converges downward to a limit $L > 0$ when $m \to \infty$. So, for any m, $\Delta_m \geq L$ and: $P^0(\mathcal{T}) \leq \frac{1}{L} \times P^\varepsilon(\mathcal{T})$.
From $\mathcal{U}_m^{\neg ok}(\mathcal{A}, T_0) \subseteq \mathcal{U}_m^{\neg ok}(\mathcal{A}^\varepsilon, T_0)$, it follows:

$$\sum_{\mathcal{T} \in \mathcal{U}_m^{\neg ok}(\mathcal{A}, T_0)} P^0(\mathcal{T}) \leq \frac{1}{L} \times \sum_{\mathcal{T} \in \mathcal{U}_m^{\neg ok}(\mathcal{A}^\varepsilon, T_0)} P^\varepsilon(\mathcal{T}).$$

When $m \to \infty$, the right part tends towards 0 (as $\mathcal{A}^\varepsilon$ converges by assumption), hence the left part tends also towards 0, which proves that $\mathcal{A}$ converges. □

From theorems 20 and 22, it follows:

Corollary 23. *Suppose that $\mathcal{L}$ is closed for original algorithm $\mathcal{A}$. If, for some sequence ε, the ε-version $\mathcal{A}^\varepsilon$ satisfies the convergence criterion (ie : $\lim_{m \to +\infty} \mathcal{E}(\mathcal{T}_m^{ok}) - \mathcal{E}(\mathcal{T}_m^{\neg ok}) + mNln(2) = -\infty$), then $\mathcal{A}$ is self-stabilizing.*

From corollary 23 and theorem 21, it follows that Herman's algorithm is self-stabilizing. This proof is completely original. By comparison, the proof in [9] was done by showing that measure (φ, d) over configurations lexicographically decreases at each step with positive probability, where φ is the number of tokens and d the minimal distance between two tokens.

8 Conclusion

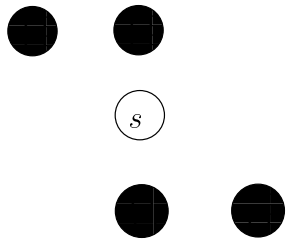
We have proposed a new method, based on principles of statistical physics, for proving the self-stabilization of a randomized algorithm $\mathcal{A}$ towards $\mathcal{L}$. Instead of showing, as classically, that $\mathcal{L}$ is the set of lowest configurations for some appropriate measure μ, we basically show that the traces of execution reaching $\mathcal{L}$ are those of minimal energy E. In contrast with μ which has to be found by hand, E is derived automatically from the form of $\mathcal{A}$. The problem is now to calculate some extreme forms of E. We have illustrated our method on Herman's example. We are able to prove similarly the self-stabilization of Flatebo-Datta's algorithm [8]. The method can be extended in principle in many ways: set of states with more than two elements, communication by reading both left and right neighbors' states, non-ring topologies (e.g.: tree, graph), $\cdots$. However these extensions complicate the calculation of the extreme energies, and application of the method to more elaborate algorithms is currently under study.

References

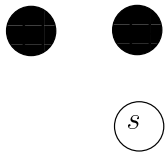
1. J. Beauquier, J. Durand-Lose, M. Gradinariu, and C. Johnen. Token based self-stabilizing uniform algorithms. *J. Parallel and Distributed Computing*, 62(5), 2002.
2. J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society*, B(31):192–236, 1974.
3. P. Brémaud. *Markov Chains, Gibbs Fields, Monte Carlo Simulation, and Queues.* Springer, 1999.
4. E.W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Communications of the ACM*, 17(11):643–644, Nov. 1974.
5. S. Dolev. *Self-Stabilization.* The MIT Press, 2000.
6. P. Duchon, P. Flajolet, G. Louchard, and G. Schaeffer. Randomized sampling from Boltzmann principles. In Springer, editor, *Proc. 29th International Colloquium on Automata, Languages and Programming, LNCS 2380*, pages 501–513, 2002.
7. M. Duflot, L. Fribourg, and C. Picaronny. Randomized distributed algorithms as Markov chains. In Springer, editor, *Proc. 15th Int. Conf. on Distributed Computing (DISC'2001), LNCS 2180*, pages 240–254, 2001.
8. M. Flatebo and A.K. Datta. Two-state self-stabilizing algorithms for token rings. *IEEE Transactions on Software Engineering*, 20(6):500–504, June 1994.
9. T. Herman. Probabilistic self-stabilization. *IPL*, 35(2):63–67, June 1990.
10. H. Kakugawa and M. Yamashita. Uniform and self-stabilizing token rings allowing unfair daemon. *IEEE Trans. Parallel and Distributed Systems*, 8(2), 1997.
11. L. Lamport. Solved problems, unsolved problems and non-problems in concurrency. In *Proc. of PODC*, pages 1–11, 1984.
12. B. Prum and J.-C. Fort. *Stochastic Processes on a Lattice and Gibbs Measures.* Kluwer Academic Publishers, 1991.

Appendix A: Form of Cliques in the Markov Field of Traces

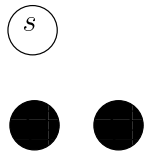
Any command of $\mathcal{A}^e$, executed at step l modifies the state of machine M_i according to the current states of M_i and its left neighbor M_{i-1}. Speaking in terms of a trace $\mathcal{T}$, this means that the state of any site $s = (l, i) \in S^*$ depends only on the states of $(l-1, i)$ and $(l-1, i-1)$. So, as neighborhood of (l, i), we should take set $\{(l-1, i-1), (l-1, i)\}$, extended with $\{(l+1, i), (l+1, i+1)\}$ (for $l \neq m$) in order to take into account the condition of symmetry met by a neighborhood (see item 2 of Def. 13). So, in case $0 < l < m$ the neighborhood of site $s = (l, i)$ is depicted by the set of black circles below:



In case $l = m$, the neighborhood of $s = (l, i)$ is just of the form:



Symmetrically, in case $l = 0$ the neighborhood of $s = (l, i)$ is of the form:



Recall that cliques are sets whose distinct elements are mutual neighbors. Accordingly, the only cliques are singletons $\{(l, i)\}$ for all $(l, i) \in S$, and pairs $\{(l, i), (l-1, i)\}$ and $\{(l, i), (l-1, i-1)\}$ for all $(l, i) \in S^*$.

Appendix B: Potentials of Clique

For any trace $\mathcal{T}$, any site $(l,i) \in S^*$ and any subset B of S, let $\Pi_{l,i}(\mathcal{T}^B)$ abbreviate for

$$\Pi_{l,i}\begin{pmatrix} \mathcal{T}^B_{l-1,i-1} & \mathcal{T}^B_{l-1,i} \\ . & \mathcal{T}^B_{l,i} \end{pmatrix},$$

with $\mathcal{T}^B_{k,j} = \mathcal{T}_{k,j}$ if $(k,j) \in B$, and $\mathcal{T}^B_{k,j} = 0$ otherwise.
(Value 0 is taken here by default, but 1 could have been taken as well.)

For each site s and each clique C^1_s (resp. C^2_s, C^3_s), a *clique potential* $V_{C^1_s}$ (resp. $V_{C^2_s}$, $V_{C^3_s}$) is defined as follows (see [3], p.264):

- For all site $s \in S^0$ (of the form $(0,i)$) and all trace $\mathcal{T}$, $\quad V_{C^1_s}(\mathcal{T}) = 0$.
- For all site $s \in S^*$, all clique C^k_s $(k = 1,2,3)$ and all trace $\mathcal{T}$

$$V_{C^k_s}(\mathcal{T}) = -\sum_{B \subset C^k_s} (-1)^{|C^k_s - B|} ln(\Pi_s(\mathcal{T}^B)),$$

where $|C^k_s - B|$ is the number of elements of C^k_s not in B.

More precisely, we have for every $s = (l,i) \in S^*$, the following potentials

$$V_{C^1_s}(\mathcal{T}) = -ln\Pi_{l,i}\begin{pmatrix} 0 & 0 \\ . & \mathcal{T}_{l,i} \end{pmatrix}$$
$$V_{C^2_s}(\mathcal{T}) = ln\Pi_{l,i}\begin{pmatrix} 0 & 0 \\ . & \mathcal{T}_{l,i} \end{pmatrix} + ln\Pi_{l,i}\begin{pmatrix} 0 & \mathcal{T}_{l-1,i} \\ . & 0 \end{pmatrix} - ln\Pi_{l,i}\begin{pmatrix} 0 & \mathcal{T}_{l-1,i} \\ . & \mathcal{T}_{l,i} \end{pmatrix}$$
$$V_{C^3_s}(\mathcal{T}) = ln\Pi_{l,i}\begin{pmatrix} 0 & 0 \\ . & \mathcal{T}_{l,i} \end{pmatrix} + ln\Pi_{l,i}\begin{pmatrix} \mathcal{T}_{l-1,i-1} & 0 \\ . & 0 \end{pmatrix} - ln\Pi_{l,i}\begin{pmatrix} \mathcal{T}_{l-1,i-1} & 0 \\ . & \mathcal{T}_{l,i} \end{pmatrix}.$$

As an example, let $\mathcal{T}$ be a trace such that $\mathcal{T}_{l,i} = 1, \mathcal{T}_{l-1,i} = 1$ and $\mathcal{T}_{l-1,i-1} = 0$, and let us compute the local potential $W_s\begin{pmatrix} 0 & 1 \\ . & 1 \end{pmatrix} = V_{C^1_s}(\mathcal{T}) + V_{C^2_s}(\mathcal{T}) + V_{C^3_s}(\mathcal{T})$ in Herman's algorithm. We have:

$$V_{C^1_s}(\mathcal{T}) = -ln\Pi_{l,i}\begin{pmatrix} 0 & 0 \\ . & 1 \end{pmatrix} = ln2$$
$$V_{C^2_s}(\mathcal{T}) = ln\Pi_{l,i}\begin{pmatrix} 0 & 0 \\ . & 1 \end{pmatrix} + ln\Pi_{l,i}\begin{pmatrix} 0 & 1 \\ . & 0 \end{pmatrix} - ln\Pi_{l,i}\begin{pmatrix} 0 & 1 \\ . & 1 \end{pmatrix}$$
$$= -ln2 + ln(1-\varepsilon_l) - ln\varepsilon_l$$
$$V_{C^3_s}(\mathcal{T}) = ln\Pi_{l,i}\begin{pmatrix} 0 & 0 \\ . & 1 \end{pmatrix} + ln\Pi_{l,i}\begin{pmatrix} 0 & 0 \\ . & 0 \end{pmatrix} - ln\Pi_{l,i}\begin{pmatrix} 0 & 0 \\ . & 1 \end{pmatrix}$$
$$= -ln2 - ln2 + ln2 = -ln2.$$

So: $W_s\begin{pmatrix} 0 & 1 \\ . & 1 \end{pmatrix} = V_{C^1_s}(\mathcal{T}) + V_{C^2_s}(\mathcal{T}) + V_{C^3_s}(\mathcal{T}) = -ln(\lambda_l) - ln2.$
The local potentials W_s for the other triples of Q^3 are computed similarly.

Appendix C: Computation of Energy $Min(\mathcal{E}_m^{\neg ok})$.

In order to get a minimal energy, we need a maximum of local potentials of value $ln(\lambda_l)$ and a minimum of value $-ln(\lambda_l)$, i.e. the maximum of patterns $\phi_0 = \begin{pmatrix} 1\ 0 \\ .\ 1 \end{pmatrix}$ and the minimum of $\phi_1 = \begin{pmatrix} 0\ 1 \\ .\ 1 \end{pmatrix}$. Before line m, the $\mathcal{L}^{\neg ok}$ trace of minimal energy is thus of the form:

$$\begin{pmatrix} 0\,1\,0\cdots 1\,0\,1\,0 \\ 1\,0\,1\cdots 0\,1\,0\,1 \\ \cdots \\ 0\,1\,0\cdots 1\,0\,1\,0 \end{pmatrix}$$

which contains $(m-1)\frac{N-1}{2}$ pattern ϕ_0 and no pattern ϕ_1. Every line contains only one token and can be derived from the previous one via original Herman's algorithm. We have:

$E(T_{l-1} \to T_l) = \frac{N-1}{2} ln(\lambda_l) - N ln(2)$, for all $1 \leq l \leq m-1$.

On the other hand, last line m cannot derive itself from line $m-1$ via Herman's original algorithm because, otherwise, the trace would be $\mathcal{L}^{ok}$. So, in lines $m-1$ and m, either a pattern $\phi_1 = \begin{pmatrix} 0\ 1 \\ .\ 1 \end{pmatrix}$ or $\phi_2 = \begin{pmatrix} 1\ 0 \\ .\ 0 \end{pmatrix}$ should appear. In the first case, the complete $\mathcal{L}^{\neg ok}$ trace is of the form:

$$\begin{pmatrix} 0\,1\,0\cdots 1\,0\,1\,0 \\ 1\,0\,1\cdots 0\,1\,0\,1 \\ \cdots \\ 0\,1\,0\cdots 1\,0\,1\,0 \\ 1\,0\,1\cdots 0\,1\,1\,1 \end{pmatrix}$$

The second case is symmetrical. In both cases, the transition energy is:

$E(T_{m-1} \to T_m) = \frac{N-3}{2} ln(\lambda_m) - (N+1) ln(2)$.

The minimal energy of an $\mathcal{L}^{\neg ok}$ trace is thus:

$$Min(\mathcal{E}_m^{\neg ok}) = \tfrac{N-3}{2} ln(\lambda_m) + \tfrac{N-1}{2} \textstyle\sum_{l=1}^{m-1} ln(\lambda_l) - (mN+1) ln(2).$$